Remarks

This Amendment responds to the final Office Action ("the Action") mailed July 25, 2007. Reconsideration of the application is respectfully requested in view of the foregoing amendments and following remarks. Claims 1-25 are pending in the application. No claims have been canceled. No claims have been allowed. Claims 1, 3, 6, 7, 14, 18, and 23 are independent.

Cited Art

U.S. Patent No. 7,140,008 to Chilimbi et al. ("Chilimbi") is entitled "Dynamic Temporal Optimization Framework."

U.S. Patent No. 7,032,217 to Wu ("Wu") is entitled "Method and System for Collaborative Profiling for Continuous Detection of Profile Phase Transitions."

U.S. Patent No. 6,658,652 to Alexander, III et al. ("Alexander") is entitled "Method and System for Shadow Heap Memory Leak Detection and Other Head Analysis in an Object-Oriented Environment During Real-Time Trace Processing."

The reference by Zorn and Hilfinger ("Zorn") is entitled "A Memory Allocation Profiler for C and Lisp Programs."

Amendments

Editorial amendments have been made to claims 1, 3, 6, 7, 14, 18, and 23. No new matter has been added.

Claim Rejections - 35 USC § 112

The Action rejects Claim 11 under 35 USC § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Specifically, the Action alleges that the term "last access" is a "relative term which renders the claim indefinite" and that "the specification does not provide a standard for ascertaining the requisite degree, and one of ordinary skill in the art would not be reasonably apprised of the scope of the invention." [Action, at § 9, page 6.] Applicants respectfully disagree with the characterization of this

language. In particular, Applicants note that the specification does indeed provide an explanation of this language, for example with respect to Figure 4:

> The memory leak detection tool 408 uses the heap allocation and free information to maintain a model of the heap, which it updates with the heap access information. *Periodically, the memory leak detection tool 408 takes a snapshot at block 410, where it visits all objects in its heap models and reports all objects that satisfy its staleness predicate as leaks. It is able to associate the responsible heap allocation, all heap frees that deallocated objects created at that allocation site, and most importantly the "last access", with each heap object reported as a leak. This last access information is invaluable for quickly debugging and fixing detected leaks. In addition, the last access information enables quick determination of "false positives".* The leak snapshots are post processed at block 412 and leak reports can then be visualized through a user terminal at block 414. The user terminal can also include a source code browser that highlights a line of code in the program that is the last access to a leaked object.

[Application, at page 11, lines 11-23; emphasis added.] Thus, applicants believe that, from at least the above-quoted language, the specification provides a standard for interpreting the language of claim 11. Applicants therefore request that the rejections be withdrawn.

Double Patenting Rejections

The Action rejects claims 1, 6, 18, and 23 under nonstatutory obviousness-type double patenting as being unpatentable over claim 1 of Chilimbi. Applicants respectfully submit the claims, as amended, are patently distinct over the cited art.

In its rejections, the Action appears to allege that the claims are each unpatentable over claim 1 alone and with no modification. [*See*, Action, at § 10, pages 6-10, where only the language of claim 1 of Chilimbi is recited against the instant claims]. For this reason, Applicants assume the Action intends to allege the claims are anticipated by Chilimbi.

Applicants respectfully submit the claims in their present form are allowable over the cited art. For an anticipation rejection to be proper, the cited art must show each and every element as set forth in a claim. However, Chilimbi does not describe each and every element.

Claims 1, 18, and 23

Applicants note that instant claims 1, 18, and 23 each recite some form of "sampling" "rates". For example, claim 1 recites:

when a code path is to be executed, determining to dispatch execution into the instrumented version code path at a sampling rate for the respective code path and otherwise into the original version code path *such that, for a given sampling rate, a ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the given sampling rate*; and

adapting the sampling rate for the code paths according to the frequency of execution of the code paths, *such that, after adapting, a ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the adjusted sampling rate.*

[Emphasis added.] Claim 18 recites:

sampling a copy of a procedure at a rate inversely proportional to how frequently either the original or the copy of the procedure is executed;

wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure.

[Emphasis added.] And claim 23 recites:

sampling a copy of a procedure at higher rates for procedures whose original versions or copies are executed less frequently and sampling a copy of a procedure at lower rates for procedures whose original versions or copies are executed more frequently;

wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure.

Examples of sampling rates, and how they are affected by execution frequency, are found in the Application, for example starting at page 6:

In bursty tracing with adaptive instrumentation, this sampling rate is adapted to the frequency of execution of the code path through the adaptive dispatch check. The more often the code path (i.e., the adaptive dispatch check) is executed, the more the sampling rate is decreased. In one implementation, all adaptive dispatch checks initially produce bursty trace samples at a rate at or near 100% (full tracing). . . .

. . .

[]For example, in one implementation, the sampling rate is decremented by a factor of 10 each time the sampling rate is decreased, e.g., from 100%, 10%, 1%, 0.1%, etc. The interval determines how often to decrement the sampling rate. In one implementation, the sampling rate is decremented progressively less often. For example, the interval between decrements can be increased by a factor of 10 each time time the sampling rate is decremented, e.g., from an interval of 10 nCheck counter resets, to 100, 1000, 10,000, etc. The bound counter determines the lower bound of the sampling rate for the adaptive dispatch check.

In each case, the Action cites either "tracking a number of iterations" or "the tracked number of iterations."

Hence, the claims recite *rates* of sampling which are, in one instance, measured in terms of percentage, such as 100%, or 1%. Furthermore, the Application describes the sampling rate in terms of ratios at page 6:

> In this bursty tracing with adaptive instrumentation, the framework maintains a per-dispatch check sampling rate rather than a global sampling rate. More specifically, a separate set of nCheck and nInstr counters are associated with each adaptive dispatch check. As in the bursty tracing framework, the nCheck counter for the adaptive dispatch check counts down a number of executions of the dispatch check that dispatch execution into the checking code. So long as the nCheck counter is non-zero, the dispatch check dispatches to the checking code 120. When the nCheck counter reaches zero, the nInstr counter is initialized to its initial value $nInstr_0$, and execution is dispatched into the instrumented version of the code 130. The nInstr counter counts down a number of executions dispatched to the instrumented code 130. When the nInstr counter again reaches zero, the nCheck counter is re-initialized to its initial value $nCheck_0$. The initial values $nCheck_0$ and $nInstr_0$ then determine the period between bursty trace samples and the length of the bursty trace, respectively. *The sampling rate (r) is then given by* $r=nInstr_0/(nCheck_0+nInstr_0)$.

[Action, at page 6, lines 10-23; emphasis added.]

In contrast, the sections of the claim 1 of Chilimbi cited against the claims does not recite "sampling rates." Instead, claim 1 of Chilimbi recites "tracking a number of iterations" of check code and "switching between checking and profiling phases" when the check code "reach[es] the respective count parameter." Additionally, the Action, in its "Response to Arguments" section, argues that this "number of iterations" language reads upon sampling rates. [*See*, Action, at § 7, page 3.]

Applicants note that a "number of iterations" does not describe a "sampling rate," as it is a simple count, whereas the rate is defined in terms of a ratio or percentage. Additionally, the "switching" language cannot describe a "sampling rate" because it is hard-coded to a count parameter and thus gives no indication that it can or will change according to a "rate." And while Applicants respectfully disagree with the assertions of the Action that this language is read upon by a "number of iterations" or "switching," Applicants have amended the claims to more clearly recite the "rate" as a ratio or percentage. For at least these reasons, Applicants argue that claim 1 of Chilimbi does not describe, nor does it teach or suggest, the "sampling" "rate" language of these instant claims.

Claim 6

Claim 6, which is also rejected in the Action, recites:

> executing the executable version of the program, wherein the copies of the procedures are executed in bursts, and the frequency at which the bursts are performed decreases as the total number of executions of either the original procedure or copy of the procedure is executed, *said frequency comprising the number of executions of the copy of the procedure divided by the total number of executions.*

[Emphasis added.] Again, similarly to the "sampling rate" argument above, Applicants note that claim 1 of Chilimbi does not recite any "frequency" of performance of bursts, "said frequency comprising the number of executions of the copy of the procedure divided by the total number of executions". The Action appears to find this language in the "switching" language of claim 1 of Chilimbi discussed above, as well as in the "upon executing the check code" language, which simply describes a mode of switching between non-instrumented and instrumented versions of code. Thus, for similar reasons to those above, claim 1 of Chilimbi does not describe, nor does it teach or suggest, the "frequency" language of instant claim 6.

For at least these reason, the Action fails to make a establish proper case of nonstatutory obviousness-type double patenting over claim 1 of chilimbi. Accordingly, applicants request that the double patenting rejection be withdrawn.

Claim Rejections under 35 USC § 102

The Action rejects claims 1-6, 14, 18, 19, and 23 under 35 U.S.C. § 102(e) as being anticipated by Wu. Applicants respectfully submit the claims in their present form are allowable over the cited art. For a 102(b) rejection to be proper, the cited art must show each and every element as set forth in a claim. (*See* MPEP § 2131.01.) However, the cited art does not describe each and every element. Accordingly, applicants request that all rejections be withdrawn. Claims 1, 3, 6, 14, 18, and 23 are independent.

Claim 1

Claim 1, as amended, recites, in part:

> when a code path is to be executed, determining to dispatch execution into the instrumented version code path at *a sampling rate for the respective code path and otherwise into the original version code path such that, for a given sampling rate, a*

ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the given sampling rate; and

adapting the sampling rate for the code paths . . ., such that, after adjusting, a ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the adjusted sampling rate.

[Emphasis added.] Examples of adapting sampling rates for the code paths were given above.

Wu's description of a "trigger counter" does not teach or suggest "adapting the sampling rate" as recited in claim 1 because the "trigger counter" merely holds a threshold for triggering an interrupt. The Action cites to Figures 5A and 5B, in particular steps 530 and 585 in its rejection of the "adapting" language of claim 1. [*See*, Action at § 12, page 12.] Applicants note, however, that the steps 530 and 585 of Figures 5A and 5B are both determination steps where it is determined if "Trigger Counter == 0." [Wu, Figures 5A and 5B.]

The "Trigger Counter" of figures 5A and 5B is a counter parameter which "invokes an interrupt after the number of overflowed profile counters has reached the trigger_counter threshold." [Wu, column 7, lines 11-12.] Indeed, Figures 5A and 5B show that, in the case that the value of trigger_counter is 0, the processes of these Figures jumps to a "Signal Phase Transition" step 535 or 590. [Wu, Figures 5A and 5B.] This phase transition means a program which is being optimized has changed from one profile phase to a new profile phase, meaning the program should be re-optimized based on the phase transition. [*See*, Wu, column 3, lines 21-25; Figures 5A and 5B, at blocks 540, 545, 595, and 599.]

Wu's determination and signaling of phase transitions, however, does not teach or suggest adapting a sampling rate. First of all, Wu does not disclose that a particular rate of sampling is performed. This is particularly true in light of the amendment, wherein the claim recites that "after adapting, [the] ratio of a number of executions of the instrumented version code path to a total number of executions of the code path is equivalent to the adjusted sampling rate." The simple "trigger counter" does not read on any such ratio.

Furthermore, because the trigger counter check cited in the rejection signals a phase transition and requires a re-optimization of code, it necessarily stops any profiling from taking place when it is tripped. While the Action, in its "Response to Arguments" suggests that such a re-optimization is "adapting," (*see*, Action, at § 7, page 4) Applicants respectfully point out that, even then, Wu gives no

indication of knowledge of a sampling rate either before or after a re-optimization. Thus, the trigger counter of Wu does not teach or suggest "adapting the sampling rate" as recited in claim 1.

For at least these reasons, Wu does not teach or suggest the above-recited language of claim 1 and thus does not describe each and every element of claim 1. Claim 1, as well as claim 2, which depends from claim 1, are thus allowable and applicants request their allowance.

Claim 3

Claim 3, as amended, recites, in part:

> []selectively reducing a frequency at which the duplicate version of the procedure is executed, *the frequency equivalent to the number of executions of the duplicate version taken as a percentage of the total number of executions of the procedure.*

[Emphasis added.] In its rejection of claim 3, the Action cites to the "decrement trigger counter" step 525 of Figure 5A of Wu. [*See*, Action at § 12, pages 12-13.] The Action also reiterates the allegation that this language reads on the "selectively reducing a frequency" language of claim 3 in its Response to Arguments. [*See*, Action at § 7, page 4.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 3 and thus does not describe each and every element of claim 3. Claim 3, as well as claims 4 and 5, which depend from claim 3, are thus allowable and applicants request their allowance.

Claim 6

Claim 6, as amended, recites, in part:

> executing the executable version of the program, wherein the copies of the procedures are executed in bursts, and the frequency at which the bursts are performed decreases as the total number of executions of either the original procedure or copy of the procedure is executed, *said frequency comprising the number of executions of the copy of the procedure divided by the total number of executions.*

[Emphasis added.] In its rejection of the above-emphasized language of claim 6, the Action cites to the "decrement trigger counter" step 525 of Figure 5A of Wu. [*See*, Action at § 12, page 14.] The Action also reiterates the allegation that this language reads on the language of claim 6 in its Response to Arguments. [*See*, Action at § 7, page 5.] For reasons similar to those discussed above with respect to claims 1 and 3, this portion of Wu discloses only indication of phase transition and thus does not

teach or suggest that "said frequency comprising the number of executions of the copy of the procedure divided by the total number of executions" as is recited in claim 6.

The rejection also cites to step 210 of Figure 2 and column 4, line 64 to column 5, line 20 of Wu, which describes compiler software which "inserts or modifies profiling instructions into the program and arranges the profile data," as well as general flows of a profile collection technique. [*See*, Action at § 12, page 14.] The Action also reiterates the allegation that this language reads on the language of claim 6 in its Response to Arguments. [*See*, Action at § 7, page 5.] However, these passages do not teach or suggest a decreasing of execution frequency, the frequency described as in claim 6, but instead focusing on collecting information and re-optimizing. [*See*, Wu at column 4, lines 64-65.] These passages thus also do not demonstrate the above-emphasized language of claim 6. Thus, Wu does not describe each and every element of claim 6. Claim 6 is thus allowable and applicants request its allowance.

Claim 14

Claim 14 recites, in part:

> executing the instrumented version of the software, wherein the additional programming code is executed more frequently when located at instrumentation points for procedures that are less frequently executed, and the additional programming code is executed less frequently when located at instrumentation points for procedures that are more frequently executed;
>
> *wherein frequency of execution of additional programming code for a given procedure comprises a number of executions of the additional programming code taken as a percentage of total executions of the procedure.*

[Emphasis added.] In its rejection of the "executing" language of claim 14, the Action cites to the "decrement trigger counter" step 525 of Figure 5A of Wu. [*See*, Action at § 12, page 15.] The Action also reiterates the allegation that this language reads on the language of claim 14 in its Response to Arguments. [*See*, Action at § 7, page 5.] For reasons similar to those discussed above with respect to claims 1 and 3, this portion of Wu discloses only indication of phase transition and thus does not teach or suggest "wherein frequency of execution of additional programming code for a given procedure comprises a number of executions of the additional programming code taken as a percentage of total executions of the procedure" as is recited in claim 14.

The rejection also cites to step 210 of Figure 2 and column 4, line 64 to column 5, line 20 of Wu. Thus, for reasons similar to those discussed above with reference to claim 6, these passages thus do not demonstrate the above-emphasized language of claim 14. Thus, Wu does not describe each and every element of claim 14. Claim 14 is thus allowable and applicants request its allowance.

Claim 18

Claim 18 recites, in part:

> sampling a copy of a procedure at a rate inversely proportional to how frequently either the original or the copy of the procedure is executed;
> *wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure.*

[Emphasis added.] In its rejection of claim 18, the Action cites to the same passages of Wu against the "sampling" language of claim 18 that were cited in the rejection of the "adapting" language of claim 1. [*See*, Action at § 12, page 15.] The Action also reiterates the allegation that this language reads on the language of claim 18 in its Response to Arguments. [*See*, Action at § 7, page 5.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 18 and thus does not describe each and every element of claim 18. Claim 18, as well as claim 19, which depends from claim 18, are thus allowable and applicants request their allowance.

Claim 23

Claim 23 recites, in part:

> sampling a copy of a procedure at higher rates for procedures whose original versions or copies are executed less frequently and sampling a copy of a procedure at lower rates for procedures whose original versions or copies are executed more frequently;
> *wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure.*

[Emphasis added.] In its rejection of claim 23, the Action cites to the same passages of Wu that were cited in the rejection of the "adapting" language of claim 1. [*See*, Action at § 12, page 16.] The Action also reiterates the allegation that this language reads on the language of claim 23 in its

Response to Arguments. [*See*, Action at § 7, page 5.] Thus, for at least the reasons discussed above with respect to claim 1, Wu does not teach or suggest at least the above-emphasized language of claim 23 and thus does not describe each and every element of claim 23. Claim 23 is thus allowable and applicants request its allowance.

Patentability of Claims 7-13 and 15-17 Under 35 USC § 103(a)

The Action rejects claims 7-13, and 15-17 under 35 U.S.C. § 103(a) over Wu in view of Alexander. The Action also rejects claims 20-22, 24, and 25 under 35 U.S.C. § 103(a) over Wu in view of Zorn. To establish a prima facie case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. [*See*, MPEP § 2142.]

Claim 7

Claim 7, as amended, recites, in part:

> executing the instrumented version of the software, wherein the instrumented version of the procedures are sampled at higher rates for procedures whose original versions or copies are executed less frequently and sampled at lower rates for procedures whose original versions or copies are executed more frequently, *wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure . . . .*

[Emphasis added.] In its rejection of the "executing" language of claim 7, the Action cites to the "decrement trigger counter" step 525 of Figure 5A of Wu. [*See*, Action at § 12, page 18.] The Action also reiterates the allegation that this language reads on the language of claim 7 in its Response to Arguments. [*See*, Action at § 7, page 5.] For reasons similar to those discussed above with respect to claims 1 and 3, this portion of Wu discloses only indication of phase transition and thus does not teach or suggest "wherein a rate for a given procedure comprises a number of executions of the instrumented version of the procedure taken as a percentage of total number of executions of either version of the procedure" as is recited in claim 7.

The rejection also cites to step 210 of Figure 2 and column 4, line 64 to column 5, line 20 of Wu. [*See*, Action at § 12, page 18.] Thus, for reasons similar to those discussed above with reference to claim 6, these passages thus do not demonstrate the above-emphasized language of claim 7. Thus, Wu does not teach or suggest this language from claim 7. Applicants do not find further such disclosure in Alexander. Thus, the combination of Wu and Alexander does not teach or suggest every element of claim 7. Claim 7 is thus allowable and applicants request its allowance.

Dependent Claims

Each of claims 8-13, 15-17, 20-22, 24, and 25 depend from claims 7, 14, 18, and 23 respectively and recite additional patentable language. In the interest of expediency, Applicants do not belabor the individual language of each claim but note that, for at least the reasons given above, Wu fails to teach or suggest each and every element of these dependent claims. Applicants do not find further disclosure in Alexander or Zorn.

For at least these reasons, the rejection of claims 8-13, 15-17, 20-22, 24, and 25 over the combinations of Wu and Alexander or Wu and Zorn is improper and fails to establish prima facie obviousness according to the standard set forth in MPEP § 2142. Thus, Applicants respectfully note that the claims are allowable. Applicants respectfully request their allowance.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

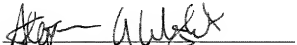
Conclusion

The claims in their present form should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By 
Stephen A. Wight
Registration No. 37,759